Ways of Giving Code to a Java Thread

Douglas C. Schmidt
d.schmidt@vanderbilt.edu
www.dre.vanderbilt.edu/~schmidt

Institute for Software Integrated Systems
Vanderbilt University
Nashville, Tennessee, USA
Learning Objectives in this Part of the Lesson

- Understand how Java threads support concurrency
- Learn how our case study app works
- Know alternative ways of giving code to a thread
  - i.e., extending the Java Thread class & implementing the Runnable interface
Ways of Giving Code to Java Threads
Ways of Giving Code to Java Threads

- Java threads *must* be given code to run
Ways of Giving Code to Java Threads

- Java threads *must* be given code to run

```java
Thread t = new Thread();
t.start();
```

*Do not use the “no argument” Thread constructor directly!!!*

Ways of Giving Code to Java Threads

- Java threads *must* be given code to run

There are alternative ways to give code to Java threads
Ways of Giving Code to Java Threads

- Java threads must be given code to run, e.g.
  1. Extend the Thread class

```java
public class GCDThread extends Thread {
    public void run() {
        // code to run goes here
    }
}
```

See docs.oracle.com/javase/8/docs/api/java/lang/Thread.html
Ways of Giving Code to Java Threads

- Java threads *must* be given code to run, e.g.
  1. Extend the Thread class

```java
public class GCDThread extends Thread {
    public void run() {
        // code to run goes here
    }
}
```

Override the `run()` hook method in the subclass & define the thread's computations

See [wiki.c2.com/?HookMethod](http://wiki.c2.com/?HookMethod)
Ways of Giving Code to Java Threads

- Java threads must be given code to run, e.g.
  1. Extend the Thread class

```java
public class GCDThread extends Thread {
    public void run() {
        // code to run goes here
    }
}
```

```java
Thread gcdThread = new GCDThread();
gcdThread.start();
```

Create & start a thread using GCDThread, which is a named subclass of Thread
Ways of Giving Code to Java Threads

- Java threads *must* be given code to run, e.g.
  1. Extend the Thread class

```java
public class GCDThread extends Thread {
    public void run() {
        // code to run goes here
    }
}
new GCDThread().start();
```

You can also write a one-liner to create & start an anonymous thread
Ways of Giving Code to Java Threads

- Java threads *must* be given code to run, e.g.
  1. Extend the Thread class
  2. Implement the Runnable interface

See [docs.oracle.com/javase/8/docs/api/java/lang/Thread.html](http://docs.oracle.com/javase/8/docs/api/java/lang/Thread.html)
Ways of Giving Code to Java Threads

- Java threads *must* be given code to run, e.g.
  1. Extend the Thread class
  2. Implement the Runnable interface

See [docs.oracle.com/javase/8/docs/api/java/lang/Runnable.html](docs.oracle.com/javase/8/docs/api/java/lang/Runnable.html)
Ways of Giving Code to Java Threads

- Java threads *must* be given code to run, e.g.
  1. Extend the Thread class
  2. Implement the Runnable interface
Ways of Giving Code to Java Threads

- Java threads *must* be given code to run, e.g.
  1. Extend the Thread class
  2. Implement the Runnable interface

```java
public class GCDRunnable implements Runnable {
    public void run() {
        // code to run goes here
    }
}

Runnable gcdRunnable = new GCDRunnable();
```

Create an instance of a named class as the runnable
Ways of Giving Code to Java Threads

- Java threads must be given code to run, e.g.
  1. Extend the Thread class
  2. Implement the Runnable interface

```java
public class GCDRunnable implements Runnable {
    public void run() {
        // code to run goes here
    }
}

Runnable gcdRunnable = new GCDRunnable();
new Thread(gcdRunnable).start();
```

Pass that runnable to a new thread object & start it
Ways of Giving Code to Java Threads

- Java threads *must* be given code to run, e.g.
  1. Extend the Thread class
  2. Implement the Runnable interface

```java
public class GCDRunnable implements Runnable {
    public void run() {
        // code to run goes here
    }
}

Runnable gcdRunnable = new GCDRunnable();
Thread.ofPlatform().start(gcdRunnable);
```

*Project Loom defines yet another way to create a Java thread*

See [download.java.net/java/early_access/loom/docs/api/java.base/java/lang/Thread.html](download.java.net/java/early_access/loom/docs/api/java.base/java/lang/Thread.html)
Ways of Giving Code to Java Threads

- Java threads must be given code to run, e.g.
  1. Extend the Thread class
  2. Implement the Runnable interface

```java
new Thread(new Runnable() {  
    public void run() {  
        // code to run goes here
    }
}).start();
```

Create & start a thread by using an anonymous inner class as the runnable
Ways of Giving Code to Java Threads

- Java threads *must* be given code to run, e.g.
  1. Extend the Thread class
  2. Implement the Runnable interface

This anonymous inner class idiom is used extensively in older Java & Android code but is tedious to program.

```java
new Thread(new Runnable() {
    public void run() {
        // code to run goes here
    }
}).start();
```
Ways of Giving Code to Java Threads

- Java threads *must* be given code to run, e.g.
  1. Extend the Thread class
  2. Implement the Runnable interface
  3. Use Java 8 lambda expressions (variant of #2)

```java
new Thread(() -> {
    // code to run goes here
}).start();
```

A lambda expression is an unnamed block of code (with optional parameters) that can be passed around & executed later

Ways of Giving Code to Java Threads

- Java threads *must* be given code to run, e.g.
  1. Extend the Thread class
  2. Implement the Runnable interface
  3. Use Java 8 lambda expressions (variant of #2)

```java
new Thread(() -> {
    // code to run goes here
}).start();
```

This approach is unwieldy if the code to run is long, complex, or needs to be used multiple times!
Ways of Giving Code to Java Threads

• Java threads *must* be given code to run, e.g.
  1. Extend the Thread class
  2. Implement the Runnable interface
  3. Use Java 8 lambda expressions (variant of #2)

Runnable r
run()
...

Runnable r = () -> {
    // code to run goes here
};

new Thread(r).start();

You can therefore store the runnable in a variable & pass it to the Thread constructor.
End of Ways of Giving Code to a Java Thread